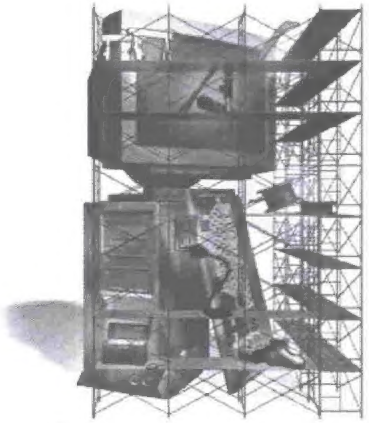
■ โยธิน อินบรรเลง



Visual FoxPro 6.0

รู้จัก ActiveX Control Microsoft ListView Control ตอนที่ 2

Microsoft ListView Control กับการกำหนดคำสั่งในโปรแกรมที่ควรรู้จักก่อนใช้งาน พร็อปเพอร์ตี้ เมธอด อีเวนต์

สวัสดีครับ พบกันเช่นเคยฉบับที่ผ่านมาเราได้รู้จักคอนโทรลชนิด OLE หรือ ActiveX ไปบ้างแล้วฉบับนี้ก็ไม่พูดพล่ามทำเพลงแล้วละสำหรับท่านที่เพิ่งเปิดมาเจอตอนที่ 2 ก็ต้องไปหาฉบับที่ผ่านมาก่อนนะครับเพราะจะทำให้เข้าใจการทำงานมากยิ่งขึ้น

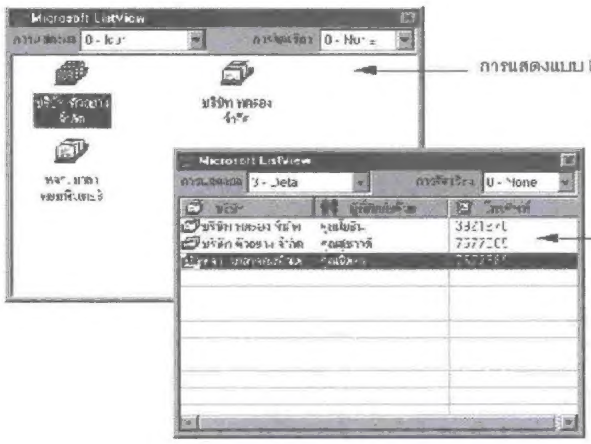
ฉบับที่ผ่านมามีคอนโทรลทั้งชนิดที่ง่าย ๆ เช่น Microsoft Clipart Gallery และที่ยากขึ้นมาอีกคือ Microsoft TreeView และ การเพิ่มรูปภาพด้วย Microsoft ImageList ฉบับนี้จะกล่าวถึง Microsoft ListView ให้ท่านผู้อ่านนำไปประยุกต์ใช้กับโปรแกรมแบบเท่ห์ อีกตามเคยครับท่าน

Microsoft ListView Control

ก่อนที่จะกล่าวถึง ListView ต้องทำให้ท่านผู้อ่านมองเห็นภาพชัดเจนก่อนว่าเราเคยเจอเจ้า List View นี้ที่ไหนบ้าง โปรแกรมที่ใกล้ตัวมากที่สุดก็คือ รูปแบบของ Windows Exploror (อีกแล้วครับท่าน) ซึ่งเป็นหน้าต่างทางด้านขวามือที่สามารถ แสดงทั้งแบบ ข้อความอย่างเดียว หรือจะแสดงทั้ง ไอคอน ด้วยแล้วแต่จะเลือกนั่นแหละครับ

รู้จักพร็อปเพอร์ตี้ เมธอด อีเวนต์

ในฉบับนี้จะกล่าวถึงการกำหนดพร็อปเพอร์ตี้อีกแบบหนึ่งที่ไม่ต้องกำหนดขณะออกแบบ (Design Time) แต่จะกำหนดผ่านคำสั่งในโปรแกรมหรือที่เรียกว่า ขณะใช้งาน (Run Time) แต่ก่อนที่จะกำหนดคำสั่งในโปรแกรมเรามาทราบถึงพร็อปเพอร์ตี้ เมธอด และอีเวนต์ ที่สำคัญในการใช้งานก่อนดังนี้



รูปที่ 1 แสดงตัวอย่างฟอร์มที่ประกอบด้วย ListView Control

Visual FoxPro 6.0

ตอนที่ 2 Microsoft LisView Control

ตารางที่ 1

พร็อปเพอร์ตี้น่าสนใจ	
พร็อปเพอร์ตี้	**รายละเอียด**
Arrange	สำหรับกำหนดการจัดเรียงไอคอนทั้งแบบปกติและขนาดเล็กสามารถกำหนดได้ดังนี้ 0 - lvwNone ไม่จัดเรียง 1 - lvwAutoLeft จัดเรียงชิดด้านซ้าย 2 - lvwAutoTop จัดเรียงให้ชิดด้านบน
CheckBoxes	กำหนดให้ใช้กล่อง CheckBox หรือไม่ (.T. = ใช้ .F. = ไม่ใช้)
ColumnHeaderIcons	กำหนดชื่อ ImageList สำหรับกำหนดไอคอนให้กับ ColumnHeaders (ต้องเพิ่ม ImageList ก่อน)
FullRowSelect	แสดงแถบสว่างในรายการที่เลือกทั้งบรรทัด (.T. = แสดง .F. = แสดงเฉพาะคอลัมน์แรก)
Icons	กำหนดชื่อคอนโทรลที่เป็น ImageList สำหรับกำหนดรูปไอคอนปกติ
SamIlcons	กำหนดชื่อคอนโทรลที่เป็น ImageList สำหรับกำหนดรูปไอคอนขนาดเล็ก
LabelEdit	กำหนดให้สามารถแก้ไขข้อความเมื่อผู้ใช้คลิกเมาส์ (.T. = แก้ไขเมื่อคลิกเมาส์ .F. = แก้ไขผ่านโปรแกรม)
Sorted	กำหนดให้ ListView จัดเรียงข้อมูลที่แสดงตามตัวชี้นำใน SortKey และจัดเรียงตามรูปแบบที่กำหนดใน SortOrder (.T. = เริ่มต้นจัดเรียง .F. = ไม่จัดเรียง)
SortKey	กำหนดหมายเลขคอลัมน์ที่ต้องการให้จัดเรียงข้อมูล (ใช้ประกอบกับ Sorted)
SortOrder	กำหนดรูปแบบการจัดเรียงข้อมูลในพร็อปเพอร์ตี้ Sorted
View	กำหนดการแสดงผลของ ListView สามารถกำหนดได้ดังนี้ 0 - lvwIcon แสดงข้อมูลแบบไอคอน 1 - lvwSmallicon แสดงข้อมูลแบบไอคอนขนาดเล็ก 2 - lvwList แสดงข้อมูลแบบรายการ 3 - lvwReport แสดงข้อมูลแบบรายงานหรือแบบแสดงรายละเอียด (Detail)

เมธอดที่เกี่ยวข้อง	
เมธอด	**รายละเอียด**
FindItem	สำหรับค้นหาข้อความหรือรายการที่กำหนดมีรูปแบบการใช้งานดังนี้ object.ListView.FindItem(string,value,index,match) ตัวอย่าง itmX = ThisForm.Lst.FindItem("หจก.",,,1) IF !ISNULL(itmX) WAIT WIND itmX.Text ENDIF พารามิเตอร์ String ข้อความที่ต้องการค้นหา Value ไม่ต้องกำหนด (ปกติจะค้นหาข้อความที่ปรากฏอยู่ใน Text) Index ไม่ต้องกำหนด Match กำหนดรูปแบบการเปรียบข้อความมี 2 รูปแบบคือ 0 - เหมือนทุกประการ 1 - เริ่มต้นบางส่วน
StartLabelEdit	แก้ไขข้อความที่เลือก

Visual FoxPro 6.0

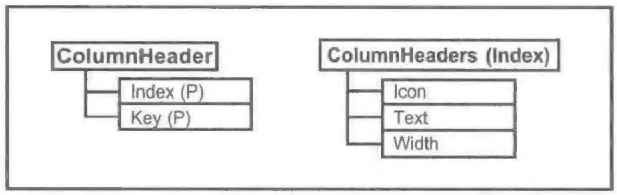
ตอนที่ 2 Microsoft LisView Control

ออปเจ็กต์ใน ListView Control

ถึงแม้ว่าใน Help จะระบุว่า ColumnHeaders, ListItems และ SelectedItemจะเป็นพร็อปเพอร์ตี้แต่หลักการโดยรวมแล้วจะถือว่าทั้งหมดต่างเป็นออปเจ็กต์หนึ่งของ ListView เพราะการอ้างอิงจะอ้างแบบออปเจ็กต์และมีทั้งพร็อปเพอร์ตี้และ เมธอด

ColumnHeader, ColumnHeaders Collection

ColumnHeader เป็นชุดออปเจ็กต์ที่ประกอบ ColumnHeader ตั้งแต่ 1 ไปจนหลาย ๆ รายการ ในส่วนนี้จะเรียกว่า ColumnHeaders Collection และการอ้างอิงกับ ColumnHeaders จะต้องกำหนดหมายเลขชี้นำ (index) ทุกครั้ง และภายใต้ ColumnHeaders จะมีรายละเอียดดังนี้



การเขียนโปรแกรมเพื่อเข้าถึงพร็อปเพอร์ตี้และใช้งานเมธอดของ ColumnHeaders จะมีรูปแบบดังต่อไปนี้

วิธีที่ 1 ไม่เก็บออปเจ็กต์

```
WITH ThisForm.Lst
   .ColumnHeaders.Add(, , "บริษัท", This.Width / 3) && เพิ่มรายการเข้ากับ ListView
   .ColumnHeaders(1).Icon = 1        && กำหนดไอคอนที่จะแสดง
         ENDWITH
```

วิธีที่ 2 เก็บออปเจ็กต์

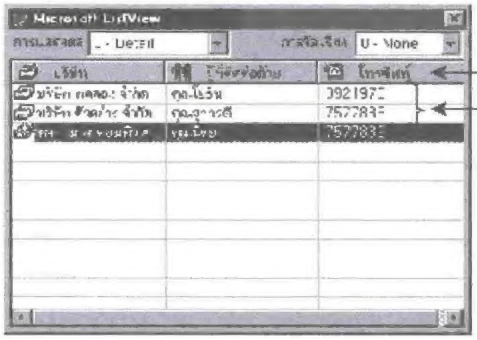
```
WITH ThisForm.Lst
      itm = .ColumnHeaders.Add(, , "บริษัท", This.Width / 3) && เพิ่มรายการเข้ากับ
ListView
      itm.Icon = 1                  && กำหนดไอคอนที่จะแสดง
ENDWITH
      -หรือ-
WITH ThisForm.Lst
      itm = .ColumnHeaders.Add( , , )        && เพิ่มรายการเข้ากับ ListView
โดยไม่กำหนดคุณสมบัติ
      itm.Icon = 1                  && กำหนดไอคอนที่จะแสดง
      itm.Text = 'บริษัท'           && กำหนดข้อความที่จะแสดง
      itm.Width = This.Width/3      && กำหนดความกว้างที่จะแสดง
ENDWITH
```

จากทั้ง 2วิธีจะเห็นว่าวิธีแรกถ้าต้องการกำหนดรายละเอียดจะกำหนดผ่าน ColumnHeaders โดยระบุตัวชี้นำโดยตรง ส่วนวิธีที่ 2 จะอ้างอิงผ่านตัวแปรที่เก็บออปเจ็กต์ ColumnHeader ที่ส่งกลับหลังจากเมธอด Add ทำงานสมบูรณ์

ListItem, ListItems Collection และ SelectedItem

ListItemและListItemsก็มีหลักการทำงานคล้ายคลึงกับ ColumnHeader และ ColumnHeaders กล่าวคือ ColumnHeaders คือ ส่วนที่แสดงหัวข้อ ส่วน ListItems คือ ส่วนที่เป็นรายละเอียดใน ListView

ส่วนSelectedItemคือรายการที่ผู้ใช้เลือกโดยการคลิกเมาส์หรือเลื่อนลูกศร จะต่างกันตรงที่ SelectedItem จะให้ค่าที่เลือกเพียงอย่างเดียวและไม่มีเมธอด Add เหมือนกับ ListItems

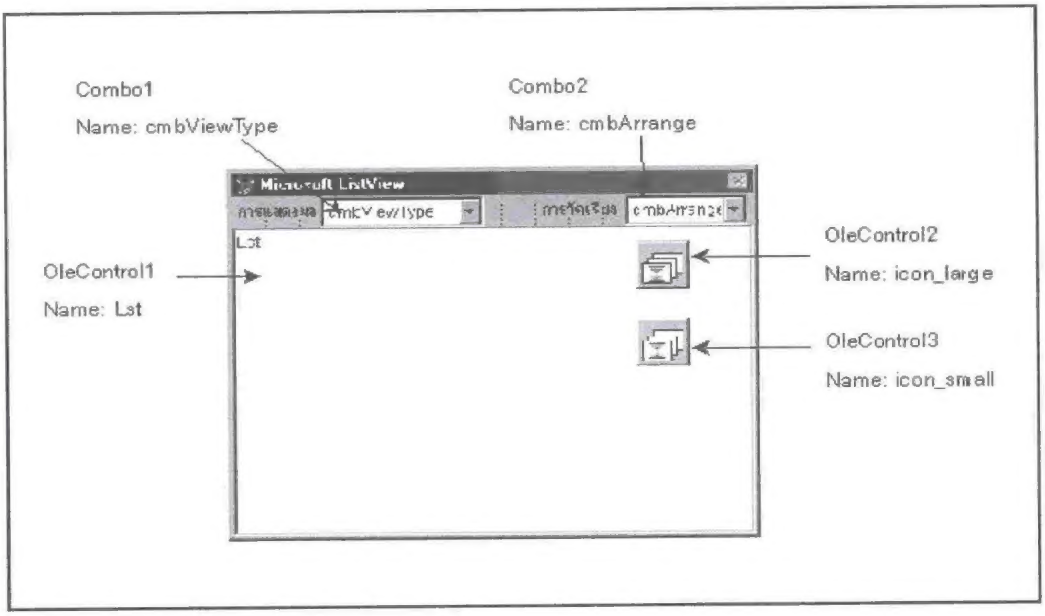


รูปที่ 2 เปรียบเทียบระหว่าง ColumnHeaders และ ListItmes

พร็อปเพอร์ตี้ใน ListItems(index)	
พร็อปเพอร์ตี้	**รายละเอียด**
Bold	กำหนดรูปแบบตัวหนาหรือปกติสำหรับข้อความใน ListItems (.T. = หนา .F. = ปกติ)
Checked	ส่งคืนหรือกำหนดค่าให้กับกล่อง CheckBoxs ที่มีอยู่ใน ListView (.T. = เลือก .F. = ไม่เลือก)
ForeColor	กำหนดสีให้กับข้อความที่แสดงใน ListView
Icon	กำหนดหมายเลขลำดับของไอคอนปกติใน ImageList
Index	ตัวเลขชี้นำของแต่ละรายการ
Key	ลำดับที่ผู้ใช้กำหนด
Selected	กำหนดหรือตรวจสอบว่ารายการใน ListItems นั้นถูกเลือกหรือไม่
Smallicon	กำหนดหมายเลขลำดับไอคอนขนาดเล็กใน ImageList
SubItems(n)	กำหนดหรือส่งคืนข้อความในรายการย่อยจะต้องระบุหมายเลขลำดับ n ด้วย
Text	ข้อความที่กำหนดให้แสดงของแต่ละรายการใน ListView

Visual FOXPro 6.0

ตอนที่ 2 Microsoft LisView Control



รูปที่ 3
แสดงตัวอย่างการสร้างฟอร์มในการใช้งาน

อีเวนต์ใน ListView

ListView คอนโทรลจะมีเหตุการณ์ตอบสนองต่อผู้ใช้ที่สำคัญอยู่ 2 รายการคือ ColumnClick และ ItemClick โดยมีรายละเอียดดังนี้

ColumnClick

เป็นเหตุการณ์เมื่อผู้ใช้คลิกเมาส์ไปยังส่วนที่เป็นส่วนหัวของตารางListViewซึ่งจะมีผลเมื่อคอนโทรลถูกกำหนดให้แสดงผลแบบ Report (หรือ Detail) เท่านั้น

เหตุการณ์นี้ทำให้เราสามารถประยุกต์ใช้ในการจัดเรียงข้อมูลตามคอลัมน์ที่เลือกได้ เช่น เมื่อผู้ใช้คลิกที่คอลัมน์ที่บรรจุชื่อบริษัท ListView จะจัดเรียงข้อมูลตามชื่อบริษัท ขณะเดียวกันถ้าผู้ใช้คลิกที่ชื่อผู้ติดต่อ ListView ก็จะจัดเรียงข้อมูลตามชื่อผู้ติดต่อ เป็นต้น

วิธีการใช้งานทำได้ง่าย ๆ เพียงดับเบิลคลิกเลือก ColumnClick ในหน้าจอ Properties แล้วกำหนดคำสั่งจัดเรียงไม่กี่คำสั่งก็ใช้งานได้แล้ว(ดูจากตัวอย่างท้ายบทความ)

ItemClick

ItemClick เป็นอีกเหตุการณ์ที่เกิดขึ้นกับ ListView เมื่อผู้ใช้คลิกเมาส์ที่รายการภายในกรอบ ListView เมื่อคลิกแล้วคอนโทรลจะส่งค่า รายการที่เลือก (Key และ Index) ไปยังอีเวนต์ ItemClick ซึ่งเราสามารถนำมาใช้งานได้ทันทีเหมือนกับ ColumnClick ส่วนการเขียนโปรแกรมพร้อมคำอธิบายจะอยู่ในตัวอย่างท้ายบทความ

ตัวอย่างการเขียนโปรแกรม

การเขียนโปรแกรมควบคุมการใช้งาน ListView ก็ไม่แตกต่างกับ TreeView คือผู้ใช้ต้องนำเอาคอนโทรลชนิด ActiveX มาไว้ในฟอร์มก่อนแล้วเลือก InsertControl/ Microsoft ListView จากหน้าจอ Insert Objects

จากนั้นให้สร้างฟอร์มโดยกำหนดรูปแบบเหมือนกับรูปต่อไปนี้

เมื่อสร้างฟอร์มเรียบร้อยแล้วให้กำหนดคำสั่งลงในแต่ละอีเวนต์ ส่วนเมธอด DspList ให้ท่านผู้อ่านสร้างใหม่จากเมนู Form/New Method แล้วกำหนดชื่อ DspList

```
Control: 1 - Form1
Events : Load
        IF USED('contact')                      && ถ้าไฟล์ Contact ถูกเปิดอยู่ให้ปิดก่อน
                USE IN contact
        ENDIF
        USE contact ORDER company IN 0          && เปิดไฟล์ Contact และจัดเรียง

Events : dsplist (สร้างขึ้นมาใหม่)
        WITH ThisForm.Lst
        .ColumnHeaders.Add(, , "บริษัท", This.Width / 3)      && แสดงข้อความที่ ColumnHeader 1
                .ColumnHeaders.Add(, , "ผู้ที่ติดต่อด้วย", This.Width / 3) && แสดงข้อความที่ ColumnHeader 2
                .ColumnHeaders.Add(, , "โทรศัพท์", This.Width / 3)  && แสดงข้อความที่ ColumnHeader 3
                .ColumnHeaders(1).Icon = 1                          && กำหนดให้แสดงไอคอนใน ColumnHeader1
                .ColumnHeaders(2).Icon = 3                          && กำหนดให้แสดงไอคอนใน ColumnHeader2
                .ColumnHeaders(3).Icon = 4                          && กำหนดให้แสดงไอคอนใน ColumnHeader3
                SELECT contact
```

Visual FOXPro 6.0

ตอนที่ 2 Microsoft LisView Control

```
                GO TOP
                DO WHILE !eof()
                        cKey = 'L'+ALLTRIM(code)          && การกำหนดคีย์จะต้องไม่เป็นตัวเลขล้วน
                        cName = ALLTRIM(company)
                        cContact = ALLTRIM(person)
                        cPhone = ALLTRIM(phone)
                        * เพิ่มรายการใน ListView โดยกำหนดคีย์สำหรับอ้างอิงด้วย Code
                        im = .ListItems.Add(,cKey,cName,)
                        im.SubItems(1) = cContact         && กำหนดให้แสดง cContact ในคอลัมน์ที่ 2
                        im.SubItems(2) = cPhone           && กำหนดให้แสดง cPhone ในคอลัมน์ที่ 3
                        im.icon = 1                       && กำหนดรูปไอคอนปกติจาก icon_large
                        im.smallicon = 1                  && กำหนดรูปไอคอนขนาดเล็กจาก icon_small
                        SKIP
                ENDDO
        ENDWITH

Control: 2 - Lst
Events: Init
WITH This.Font                  && กำหนดรายละเอียดฟอนต์ให้กับ ListView
        .Name = 'Ms Sans Serif'
        .Size = 10
ENDWITH
WITH This
        .HideSelection = .F.    && ไม่ซ่อนแถบเลือกเมื่อออกจากคอนโทรล ListView
        .FullRowSelect = .T.    && กำหนดให้แสดงแถบเลือกเต็มบรรทัด
        .Gridlines = .T.        && แสดงเส้นตารางใน ListView
        .LabelEdit = 1          && การแก้ไขข้อความต้องแก้จากโปรแกรมเท่านั้น
        .Icons = ThisForm.icon_large        && กำหนด ImageList ให้กับ Icon ทั่วไปของ ListView
        .SmallIcons = ThisForm.icon_small   && กำหนด ImageList ให้กับ Small Icon ของ ListView
        ThisForm.DspList                    && เรียกเมธอด DspList ของฟอร์มเพื่อแสดงรายการใน ListView
ENDWITH


Events : ColumnClick
        *** ActiveX Control Event ***
        LPARAMETERS columnheader
        * เมื่อผู้ใช้คลิกที่ Column Header (กรณีที่เลือกการแสดงแบบ Detail) จะสั่งให้ ListView จัดเรียงข้อมูลตาม
        * หมายเลขชี้นำใน ColumnHeader และต้องลบด้วย 1 เพราะตัวเลขชี้นำจะเริ่มด้วย 1 ส่วนการจัดเรียงจะเริ่มจาก 0
        This.SortKey = ColumnHeader.Index-1
        This.Sorted = .T.               && กำหนดให้เรียงข้อมูล

Events : ItemClick
*** ActiveX Control Event ***
LPARAMETERS item
WAIT WIND item.key NOWA

Events : KeyPress
        *** ActiveX Control Event ***
        LPARAMETERS keyascii
        IF keyascii = 13                && ถ้าผู้ใช้กดปุ่ม Enter ให้เรียกอีเวนต์ DblClick ของ ListView
                This.DblClick
        ENDIF

Events : DblClick
        *** ActiveX Control Event ***
        *** เมื่อผู้ใช้ดับเบิลคลิกที่รายการใน ListView โปรแกรมจะเก็บรายละเอียดที่ต้องการแล้วแสดงในกรอบข้อความ
        WITH This
                nIdx = .SelectedItem.Index          && เก็บหมายเลขชี้นำของรายการที่เลือก
                cIndex = ALLTRIM(STR(nIdx))         && แปลงให้เป็นตัวอักษรเพื่อแสดงใน MESSAGEBOX
```

Visual FoxPro 6.0

ตอนที่ 2 Microsoft LisView Control

```
                cKey = .ListItems(nIdx).Key          && เก็บค่า Key ของรายการที่แสดงเลือก
                cText = .ListItems(nIdx).Text        && เก็บข้อความของรายการที่เลือก
                cSeek = SUBS(cKey,2)   && กำหนดค่าสำหรับการค้นหาข้อมูลในไฟล์ กรณีนี้จะเรียงตาม Code
                =MESSAGEBOX('Index = '+cIndex+CHR(13)+ ;
                            'Key = '+cKey+CHR(13)+ ;
                            'Text = '+cText+CHR(13)+ ;
                            'Seek Expr. = '+cSeek, ;
                            0+64+0,'Your are selected.')
        ENDWITH

Control: 3 - cmbViewType
Events : Init
        * เริ่มต้นตอนโปรแกรมทำงานจะเพิ่มรายการเข้าไปใน ComboBox สำหรับกำหนดชนิดการแสดงผล
        This.AddItem('0 - Icon')
        This.AddItem('1 - Small Icon')
        This.AddItem('2 - List')
        This.AddItem('3 - Detail')
        This.DisplayValue = '0 - Icon'          && กำหนดแสดงค่า 0 - Icon ใน ComboBox

Events : InteractiveChange
        * เมื่อมีการเปลี่ยนแปลงข้อมูลใน ComboBox โปรแกรมจะนำเอาค่าตัวเลขที่แสดงอยู่ใน ComboBox มาแปลงแล้ว
        * นำค่าตัวเลขนั้นไปกำหนดรูปแบบการแสดงผลของ ListView
        vwtype = VAL(LEFT(ThisForm.cmbViewType.DisplayValue,1))
        WITH ThisForm.Lst
                .View = vwtype
                .SetFocus
        ENDWITH

Events : Click
        * ถ้าผู้ใช้คลิกไปที่ ComboBox โปรแกรมจะย้ายโฟกัสไปยัง ListView ทันที
        ThisForm.lst.SetFocus

Control: 4 - cmbArrange
Events : Init
        * เริ่มต้นตอนโปรแกรมทำงานจะเพิ่มรายการเข้าไปใน ComboBox สำหรับกำหนดการจัดตำแหน่งของไอคอน
        This.AddItem('0 - None')
        This.AddItem('1 - Left')
        This.AddItem('2 - Top')
        This.DisplayValue = '0 - None'          && เริ่มต้นให้แสดงค่า 0 - None

Events : InteractiveChange
        * เมื่อมีการเปลี่ยนแปลงข้อมูลใน ComboBox โปรแกรมจะนำเอาค่าตัวเลขที่แสดงอยู่ใน ComboBox มาแปลงแล้ว
        * นำค่าตัวเลขนั้นไปกำหนดรูปแบบการจัดตำแหน่งของไอคอนใน ListView (เฉพาะการแสดงผลแบบไอคอนเท่านั้น)
        vwtype = VAL(LEFT(ThisForm.cmbArrange.DisplayValue,1))
        WITH ThisForm.Lst
                .Arrange = vwtype
                .SetFocus
        ENDWITH

Events : Click
        * ถ้าผู้ใช้คลิกไปที่ ComboBox โปรแกรมจะย้ายโฟกัสไปยัง ListView ทันที
        ThisForm.Lst.SetFocus
```

■

ส่งท้าย

จากตัวอย่างที่ผ่านมาผู้เขียนหวังเป็นอย่างยิ่งที่จะจุดประกายให้ท่านผู้อ่านสามารถพัฒนาและประยุกต์ต่อไป เพราะการทำงานของ Microsoft ListView ไม่ได้จำกัดเพียงเท่าที่เห็นในบทความนี้เท่านั้นท่านผู้อ่านสามารถศึกษาต่อจาก MSDN Library ถึงจะหายากสักเล็กน้อยแต่ก็ไม่เหนือกว่าความพยายามของท่าน

เป็นอันว่าตอนนี้เราทราบถึงคอนโทรลที่เป็น ActiveX ถึง 4 รายการแล้วฉบับหน้าจะกล่าวถึง ToolBar ที่สามารถเพิ่มรูปภาพสวย ๆ ได้และใช้งานง่าย ๆ โปรดคอยติดตามนะครับ...